Variable and Services Querying

Goal

Provide sufficient query and facet APIs to enable variables and services to be queried as first class search terms, just like collection level search terms.

Traceability

Background

EDSC implemented a prototype workflow based on the End-To-End Services whitepaper. The CMR now needs to implement an API layer to support the prototype workflow.

Key User (Client) Stories

As a client, I want to see a list of the top measurements. (Facets v2) **CMR-4078**

As a client, I want to search through a list of measurements by measurement name to narrow it down to a smaller list. **CMR-4079**

As a client, I want to select a measurement and see all associated variables (Facets v2) **Combined with CMR-4078**

As a client, I want to search through a list of variables by variable name to narrow it down to a smaller list. **CMR-4081**

As a client, I want to search for collections based on a list of measurement names (*see outstanding question below regarding AND/OR) **CMR-4082**

As a client, I want to search for collections based on a list of variable names **CMR-4083**

As a client, I want to see a list of service options (--- need to decompose this further) **CMR-4086**

As a client, I want to search for collections based on a list of service options **CMR-4087**

As a client, I want to search for granules based on a list of measurement names * **CMR-4084**

As a client, I want to search for granules based on a list of variable names **CMR-4085**

Also wrote CMR-4080 to support the client requesting a number of values to be returned for a given facet.

Questions

- If multiple measurements are added to the query parameters, should they be AND'd or OR'd? **Answer:** OR
- What service options should be queryable? (Need to flag them on the UI and then ensure we map those to UMM-S fields.)
- Are measurements and variables two separate concepts with measurements as a parent and variables as a child (analogous to

collections and granules)? Note that the design assumes we are only saving variable concepts which include a measurement as part of their metadata.

- From my latest talk with Simon Cantrell, Variables, Measurements and Services are all collection level. We will not have them associated with granules. So in the designs, I am not considering the granules association. But I will keep the existing granule reference in this document until we are absolutely sure about it,
- Are variable names guaranteed to be unique? We assumed they are. Is that a reasonable assumption? **Answer:???**

Potential Pitfalls

1. Facets code fragmentation - We have two different ways of representing facets in the EDSC (the sidebar and the variables modal). Ideally we would come up with a good way to expose facets and provide that same behavior for all facets.
   a. Less for a user to have to learn in order to effectively use EDSC
   b. Less code to maintain (both EDSC and CMR)
2. Tight coupling between EDSC and CMR for Variables
   a. Related to the facets fragmentation. We want CMR to be responsible for the facets content (EDSC should not have to write any code to handle a new facet or know that CMR is adding or removing a facet). This was the entire driver between v2 facets.
3. We will need to do a lot of rework in a following PI in order to scale variables to support potentially hundreds of millions of granules meaning that a lot of time spent this PI will have been wasted for the end solution.

CMR Design

Measurements / Variables

Measurements and Variables are represented by a new UMM-Variables concept. See https://git.earthdata.nasa.gov/projects/EMFD/repos/unified-metadata-model/browse/DRAFT/umm-var-json-schema.json. The long-name of Variable is the name of the Measurement that it is associated with.

Metadata-db

We will save UMM-Variables concepts as immutable revisions similar to all other concepts in the CMR. Variables can map to multiple collections and multiple providers (TODO: Confirm this is true). There are not a significant number of measurements and variables (far less than 1 million) so we can just have a single CMR_VARIABLES table in metadata-db (TODO: Confirm CMR_VARIABLES is the name we want and not CMR_MEASUREMENTS).

We would have the same columns as our other concept tables:

ID
CONCEPT_ID
NATIVE_ID
METADATA
FORMAT
REVISION_ID
REVISION_DATE
CREATED_AT
DELETED
USER_ID
TRANSACTION_ID

The prefix we will use for concept-ids will be "V".

We need to be able to associate collections and granules with variables. That means that we can have hundreds of millions of associations. For the purposes of the current PI (PI-5), we will have a small number of associations. We should investigate the difficulty of reusing the CMR_TAG_ASSOCIATIONS table to track associations. Perhaps we could repurpose the tag-key to be the native-id for the variable. native-id should be the name of the Variable that is defined in its metadata and it needs to be unique across the CMR for each Variable. If we determine that there are any problems reusing the tag associations table we will need to add a new CMR_VARIABLE_ASSOCIATIONS table.

We will need an internal API for searching for variable associations for a provided collection or granule concept ID.

Ingest

We'll need an API that takes a UMM-Variables concept and saves it in metadata-db. Ingest is the logical location for this endpoint. Ingest should perform minimal validation during PI-5 (we may want to define our own simplified UMM-Variable schema that only requires the fields needed to support the EDSC prototype). Those fields would be a Measurement name, human readable measurement name, and an array of Variables, in which each Variable would be a variable name and a human readable variable name.

After validating the concept ingest would save the concept revision to metadata-db.

Ingest will need to validate a user has permission to create a concept. We should reuse the same permission check for creating system level tags for now.

Indexer

Indexer will need to index measurements and variables within collections and granules within Elasticsearch. We will not have a separate index for measurements and variables, rather they will be fields stored within the collection and granule indexes. When a collection or granule is indexed we will perform a query to find any associated variables (analogous to tag associations) and index those as part of the collection or granule. In addition when a new variable association is created we will index the associated collection or granule. Similarly when deleting variable associations we will reindex the previously associated collection or granule to remove the association.

We will index measurements and variables as a single nested field similar to science keywords. We'll need to be able to retrieve them hierarchically for supporting V2 facets (again similar to hierarchical science keywords).

When a Variable is updated, we will find all collections that are associated with the Variable and re-index those collections. This is different from how we process regular tags (we don't reindex any collections when tags are updated). We do this mostly to update the services that are associated with the collection rather than update the variable itself.

Search

The bulk of the changes to support Variables will be in the search application. The changes include:

1. Creating associations
    a. Between variables and collections.
    b. Between variables and granules.
2. Searching for collections by measurements
    a. Via new query parameter on the query parameter API
    b. Via JSON query
3. Searching for granules by measurements
4. Searching for collections by variables
    a. Via new query parameter on the query parameter API
    b. Via JSON query
5. Searching for granules by variables
6. Return Variables in V2 facets

Creating Associations

We will create associations using the existing tag associations API. In place of the tag-key we will instead use the variables concept's native ID. We will need to add a type to tag (without a type means general tags, type "variable" means the tag is a variable). We'll need to update any code that expects tag associations are always for tags.

Searching for collections by measurements

1. We will add a new collection query parameter named measurement. It will allow multiple values which by default are OR'ed together. The 'pattern' and 'AND' options will be supported, but not ignore_case as case sensitivity is not important for distinguishing between measurements.
2. Similarly we will add a new field in the JSON Query Language schema called measurement.

Searching for granules by measurements

We will add a new granule query parameter named measurement. It will allow multiple values which by default are OR'ed together. The 'pattern' and 'AND' options will be supported, but not ignore_case as case sensitivity is not important for distinguishing between measurements.

Searching for collections by variables

1. We will add a new collection query parameter named variable. It will allow multiple values which by default are OR'ed together. The 'pattern' and 'AND' options will be supported, but not ignore_case as case sensitivity is not important for distinguishing between variables.
2. Similarly we will add a new field in the JSON Query Language schema called variable.

Searching for granules by variables

We will add a new granule query parameter named variable. It will allow multiple values which by default are OR'ed together. The 'pattern' and 'AND' options will be supported, but not ignore_case as case sensitivity is not important for distinguishing between variables.

Return Variables in V2 facets

We will need to support variables as a hierarchical V2 facet similar to science keywords. In addition to all of the current V2 facets requirements we will also need a way to retrieve more than 50 values for a given facet (and a way to request a value other than the default of 50).

Services

Services are associated with collections through Variables. According to Simon Cantrell, Service's JSON schema will be updated to include the Variables that are associated with it.

There will be a small number of services in CMR, so we should update our current METADATA_DB schema to delete all of the provider specific services tables and just have a single CMR_SERVICES table.

Services schema: https://git.earthdata.nasa.gov/projects/EMFD/repos/unified-metadata-model/browse/DRAFT/umm-s-json-schema.json

Questions

- Does a collection must contain all the Variables defined in a service to be associated with the Service? OR any one of the Variables or some of them on a case by case basis?

Metadata-db

Here are the columns of the CMR_SERVICES table:

ID
CONCEPT_ID
NATIVE_ID
METADATA
FORMAT
REVISION_ID
REVISION_DATE
CREATED_AT
DELETED
ENTRY_ID
ENTRY_TITLE
DELETE_TIME
USER_ID
TRANSACTION_ID

Services are saved as a system level concept in metadata-db with its associated variables in the metadata field. There are limited amount of services in CMR and they will be cached in consistent cubby cache so that CMR can quickly search for related services for a given variable. The cache will hold a service-to-variables mapping (which is a service-name to its associated Variables mapping) and a variable-to-services mapping (which is variable-name to its associated services mapping). This cache is populated by loading the latest services from metadata-db and parsing it. Whenever a service is added/updated/deleted, the cache is refreshed immediately at the end of the add/update/delete operation.

In the first cut, we only support searching Services by its name and associated Variables.

Deletion/force-deletion: There is no cascading deletion relationship between collections, variables, measurements and services.

CMR will add validation to not allow deletion of variables that are referenced by a service

Deletion of variables and services will cause reindexing of related collections.

Deletion of collection will not affect the indexing of variables and services

Ingest

We'll add ingest endpoint to ingest a UMM-Services concept and saves it in metadata-db. Ingest should perform minimal validation during PI-5 (we may want to define our own simplified UMM-Service schema that only requires the fields needed to support the EDSC prototype). Those fields would be a Service name, human readable Service name, and an array of Variables names that are associated with the Service.

After validating the concept, ingest would save the concept revision to metadata-db.

Ingest will need to validate a user has permission to create a concept. We should reuse the same permission check for creating system level concepts for now.

Indexer

Services are indexed in its own elasticsearch mapping to allow discovery of the Services and their associated Variables.

Services to collection association is indexed in the collection mapping as a "services" field (nested field) which saves the list of Services (TBD the subfields of the Service object) that are associated with the collection.
When indexer gets a service concept update event, not only the Service is indexed in its own service mapping, the union of variables associated with the service concept and its previous revision are found; and any collections that are associated with those variables will be re-indexed as well.

During indexing of a collection, associated tags are retrieved. General tags are indexed on the collection like before. variable tags will be processed to find measurements and services that are associated with them. Then variables, measurements, services are indexed on the collection as additional fields.

Search

We will update search to add support to:

1. Searching for collections by Services:
   a. Via new query parameter on the query parameter API
   b. Via JSON query
2. Return Services in V2 facets

Order to work tickets

TBD

Epic

CMR-4009 - Getting issue details... STATUS

Tickets

Type	Key	Summary	Assignee	Reporter	Priority	Status	Resolution	Created	Updated	Due

⚠ JIRA project doesn't exist or you don't have permission to view it.

View these issues in JIRA

Error rendering macro 'pageapproval' : null